Санкт-Петербургский государственный университет
**Кафедра теории систем управления электрофизической аппаратурой**

## Ращенко Дмитрий Владимирович

**Магистерская диссертация**

# Методы обучения искусственных нейронных сетей в задачах распознавания образов

Направление 010900
«Прикладные математика и физика»
Магистерская программа «Информационные и ядерные технологии»

Научный руководитель,
кандидат физ.-мат. наук,
доцент
Козынченко В. А.

Санкт-Петербург
2018

# Оглавление

Введение . . . . . 3

Постановка задачи . . . . . 10

Проблематика и обзор литературы . . . . . 10

**1 Описание экспериментов** **14**

1.1 Выбор нейронной сети, алгоритма обучения и других параметров . . . . . 15

1.2 Выбор метрик . . . . . 19

**2 Программная реализация, тестирование и анализ результатов** **22**

2.1 Инфраструктура . . . . . 22

2.2 Результаты тестирования, выбор метрик . . . . . 23

2.3 Эволюция выбранных параметров во время обучения . . . . 25

2.4 Анализ результатов . . . . . 27

**3 Применение разработанных критериев и перспективы** **30**

3.1 Динамическое расширение слоя . . . . . 31

3.2 Многослойные сети . . . . . 31

Заключение . . . . . 33

# Введение

В последние годы наблюдается активной рост интереса к искусственным нейронным сетям (ИНС) как к методу решения многих практических задач. Этот интерес связан с разработкой новых архитектур и методов обучения, которые позволили преодолеть основные проблемы ИНС и значительно расширить круг решаемых ими задач. Также актуальность исследований нейронных сетей дополняется стремительным увеличением вычислительных мощностей, доступных исследователям, а в частности распространением графических процессоров, которые в последнее время стали основным средством для обучения сложных ИНС.

Такие авторы как Джеффри Хинтон и Ян Лекун своими работами (например, [1, 2]) внесли большой вклад в процесс выхода нейросетевых исследований из зимы (период с 1990 по 2010 гг. [3]). Описанные в этих и последующих работах методы позволили обучать действительно глубокие нейронные сети (ранее ИНС с 8 слоями считалась очень глубокой, в то время как современная сеть ResNET использует 150 слоев [4], и это не предел) для более глубокого анализа информации. Также стало возможно эффективное обучение рекуррентных ИНС, позволяющих обрабатывать временные ряды данных (например аудио, видео, тексты).

Столь стремительное расширение возможностей нейронных сетей требует проведения дополнительных исследований по выявлению применимости данного подхода к той или иной задаче, разработке новых архитектур и преодолению существующих ограничений. Подобные исследования позволят эффективно использовать столь мощный инструмент как ИНС и предотвратить наступление еще одного периода застоя.

Одной из важнейших проблем, возникающих при проектировании ИНС, является выбор ее архитектуры (глубина, ширина, тип слоев нейронов). Для каждой задачи требуется индивидуальный подход, и часто выбор архитектуры сводится к обучению множества нейронных сетей с различной архитектурой в поисках оптимальной. Поскольку обучение даже одной нейронной сети является вычислительно дорогим (требуется оптимизация

градиентными методами до миллиарда параметров), такой перебор представляется очень неэффективным. В рамках данной работы проводится исследование и разработка методов для оптимизации выбора количества нейронов в полносвязном слое ИНС.

## Введение в теорию искусственных нейронных сетей

Исследование искусственных нейронных сетей началось с работ У. Маккаллока и У. Питса на стыке нейробиологии и математики [5], в которых они предложили математическую модель биологического нейрона как взвешенного порогового сумматора входных сигналов. Нейрон такого типа изображен на рисунке 1, и результат его работы можно представить в виде $y = F(\vec{x} \cdot \vec{w})$, где $\vec{x} = (x_1, x_2, ..., x_n)$ — вектор входных сигналов, $\vec{w} = (w_1, w_2, ..., w_n)$ — строка-вектор весовых коэффициентов, $F : \mathbb{R} \to \mathbb{R}$ — функция нелинейного преобразования. Функцию $F$ принято называть активационной, и в случае нейрона Маккаллока-Питса она равна

$$F(z) = \begin{cases} 1, \text{если } x > 0; \\ 0, \text{если } x \leqslant 0 \end{cases} \tag{1}$$

Позже были разработаны и другие модели, более точно повторяющие

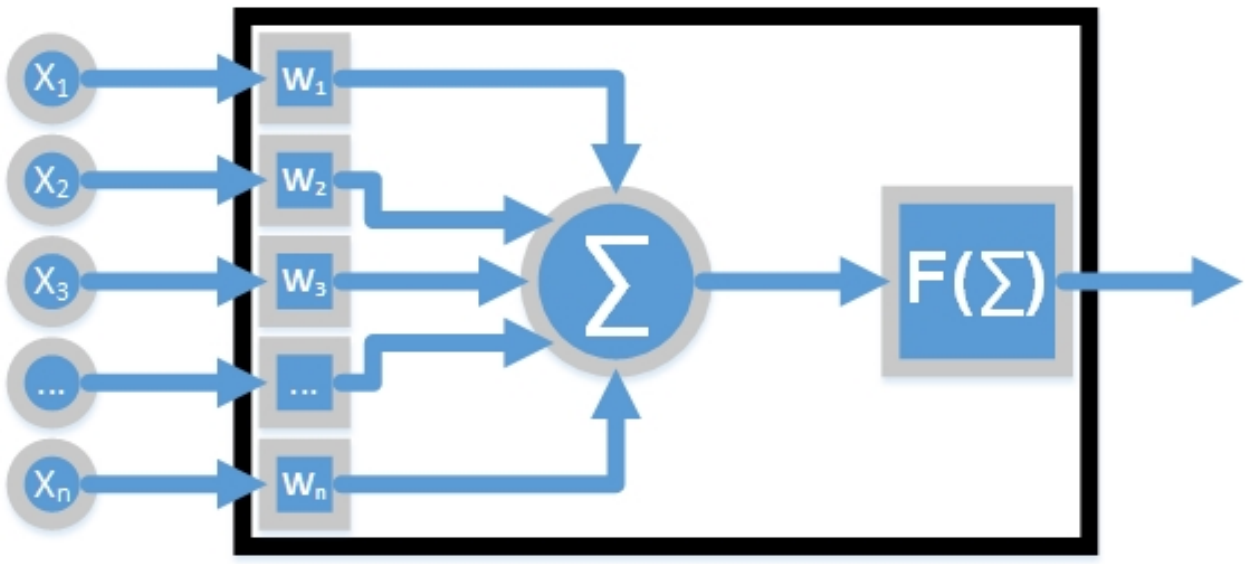


*Рис. 1.* Математическая модель нейрона Маккаллока-Питса.

свойства нейронов, но именно простые модели типа Маккаллока-Питса с некоторыми изменениями до сих пор широко используются в ИНС. При правильном выборе весовых коэффициентов нейрон такого типа может использоваться для линейного разделения $n$-мерного пространства, а объединение нескольких нейронов (путем подачи выхода одного нейрона на вход другого) может использоваться в задачах классификации и аппроксимации. Итерационный процесс выбора весовых коэффициентов (оптимизации) принято называть обучением.

На сегодняшний день наиболее распространена следующая модель нейрона:

$$y = F(\vec{x} \cdot \vec{w} + b), \tag{2}$$

не сильно отличающаяся от модели Маккаллока-Питса. Здесь $b$ — дополнительное смещение (bias) полученной суммы, которое применяется в некоторых задачах. $b$ не является фиксированным параметром и также подвергается оптимизации при обучении. В качестве активационной функции $F$ на данный момент распространены:

- функции сигмоидального типа (сглаженные пороговые). В качестве такой функции часто применяется

$$F(z) = \sigma(z) = \frac{1}{1 + e^{-z}} \tag{3}$$

  и гиперболический тангенс;

- функции типа ReLU (rectified linear unit). За основу таких функций взята $F(z) = \max(0, z)$, к которой могут применяться различные модификации: сглаженный ReLU $F(z) = \log(1 + \exp(z))$, ReLU с утечкой $F(z) = \max(z, az), a < 1$ и другие;

- в задачах классификации распространено использование функции softmax

$$F_i(z_1, z_2, ..., z_K) = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}}, i = 1, 2, ..., K \tag{4}$$

  для сравнения выходов $K$ нейронов. В результате применения softmax

выходы всех участвующих нейронов находятся в интервале от 0 до 1, а их сумма равна 1. Использование такой активационной функции является расширением описанной модели нейрона, так как подразумевает взаимную зависимость выходов нескольких нейронов.

Столь простые вычислительные элементы как нейроны могут давать интересные результаты при объединении их в сети и применении различных алгоритмов обучения. Первой нейронной сетью, которая могла распознавать визуальные образы стал перцептрон Розенблата [6], изображенный на рисунке 2.

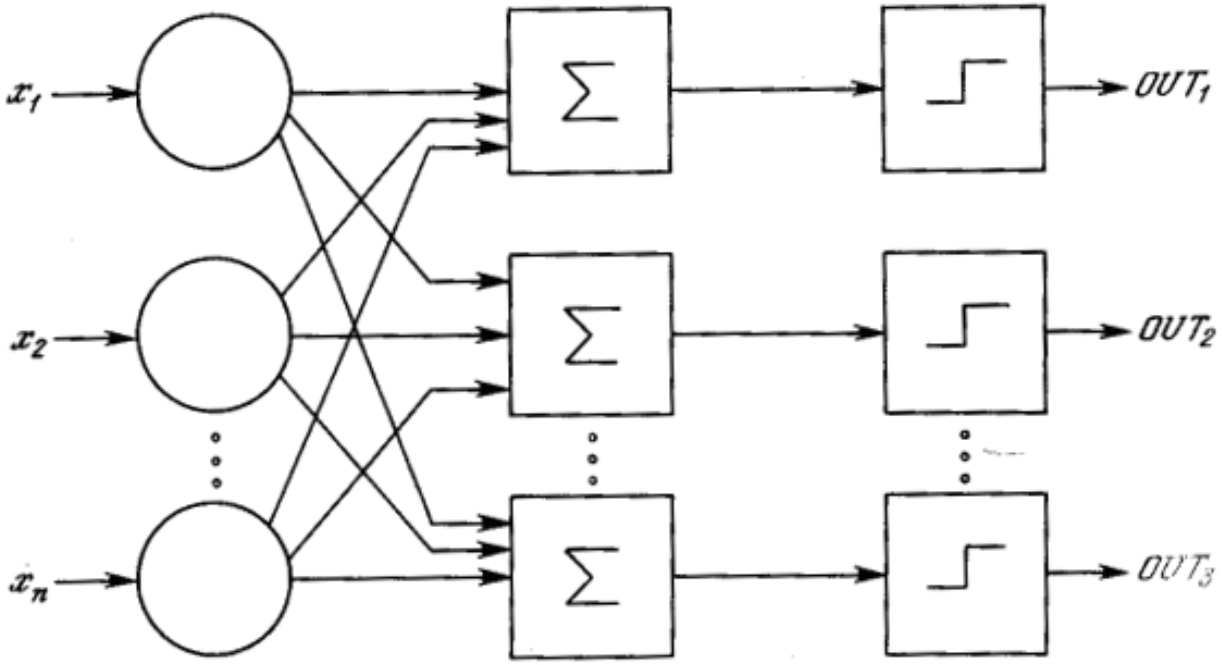


*Рис. 2.* Однослойный перцептрон Розенблата со многими выходами [7].

Со времен этой нейронной сети было разработано множество методов соединения нейронов в сети, среди которых можно выделить два класса: сети прямого распространения и сети, имеющие обратные связи (рекуррентные). В процессе прохождения сигнала от входа сети прямого распространения к выходу, каждый нейрон производит вычисления только один раз. Пример такой ИНС приведен на рисунке 3.

В рекуррентных нейронных сетях сигнал может образовывать циклы, проходя несколько раз через один нейрон. На рисунке 4 приведен пример сложной рекуррентной нейронной сети АРТ со множественными обратны-

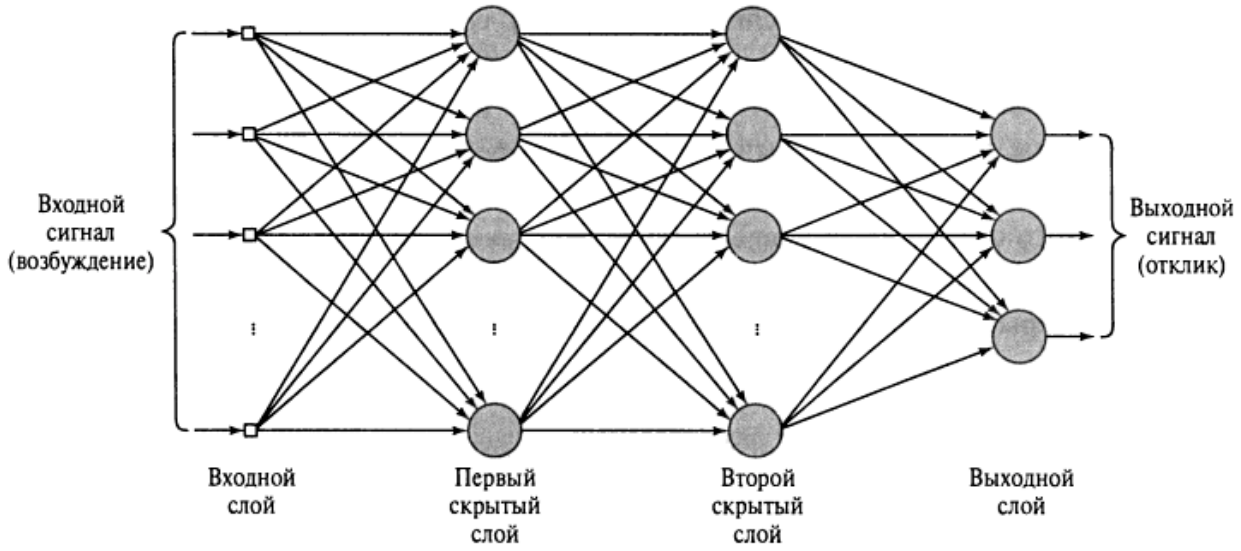


*Рис. 3.* Архитектурный граф многослойного персептрона с двумя скрытыми слоями [8].

ми связями [9]. Для простоты в данной работе будут рассматриваться только нейронные сети прямого распространения, однако рекуррентные также очень эффективны в некоторых задачах и широко используются [11, 12].

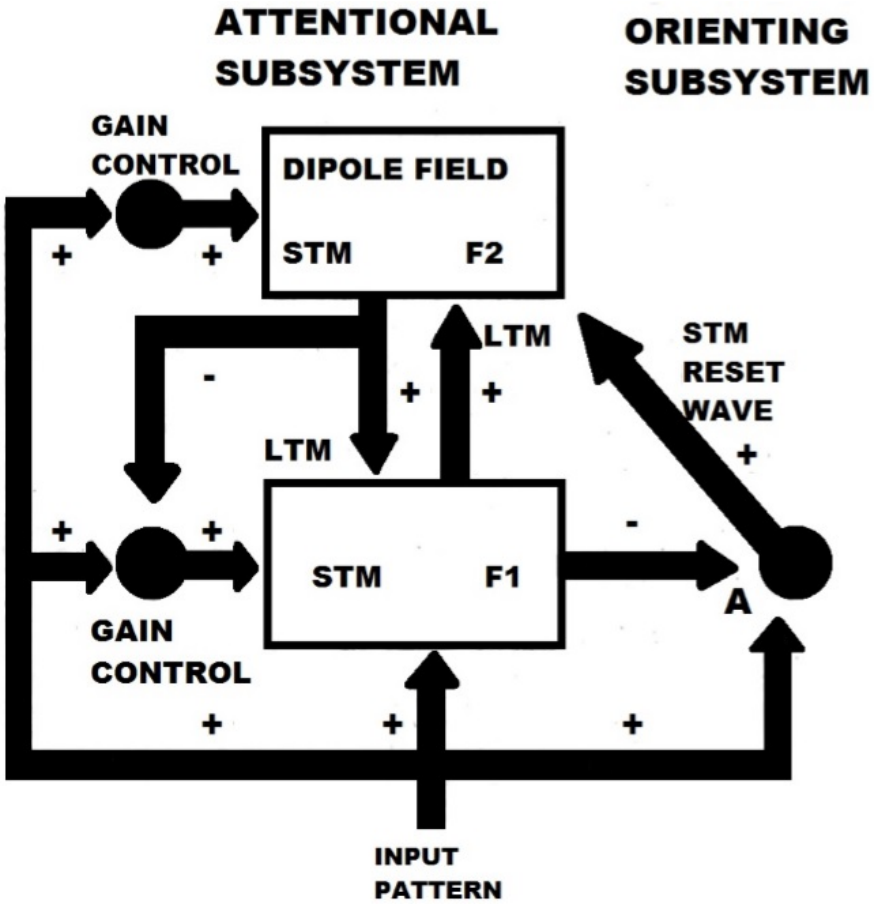


*Рис. 4.* Схема нейронной сети с обратными связями APT-1 [10].

Еще одним широко распространенным приемом упрощения нейронных сетей является объединение схожих нейронов в группы. В частности, если несколько нейронов имеют одно множество входных сигналов и одну активационную функцию, такую группу называют нейронным слоем. Результат работы слоя из $m$ нейронов можно представить в виде матричного умножения

$$\vec{y}_{[m]} = \vec{F}(\vec{x}_{[n]} \cdot W_{[m \times n]}). \tag{5}$$

Здесь $\vec{y} = (y_1, y_2, ..., y_m)$ — выходной вектор-строка, элементами которой являются выходы нейронов, входящих в слой; $W = (\vec{w}_1, \vec{w}_2, ..., \vec{w}_m)^T$ — весовая матрица, строками которой являются весовые вектор-строки входящих в слой нейронов; $\vec{F}(\vec{z}) = (F(z_1), F(z_2), ..., F(z_m))$ — функция отображающая $\vec{F} : \mathbb{R}^m \to \mathbb{R}^m$ путем поэлементного применения одной скалярной функции $F$ к каждому элементу аргумента; для удобства размерности векторов и матриц приведены в квадратных скобках. Вычисленный таким образом выход $\vec{y}$ передается на вход следующему слою. Матричный подход широко используется в ИНС, поскольку облегчает вычисления, анализ и проектирование нейронных сетей. Благодаря этим преимуществам объединение нейронов в слои получило очень широкое распространение и применяется практически во всех современных ИНС (как прямого распространения, так и рекуррентных).

Использование концепции нейронных слоев позволяет перейти от нейрона как элементарной единицы нейронной сети к нейронному слою как основному структурному элементу. Таким образом, проектирование архитектуры нейронной сети прямого распространения сводится к выбору количества последовательно идущих слоев (глубина), количества нейронов в каждом слое (ширина слоя) и типа слоя. Например, могут использоваться следующие типы слоев:

- может варьироваться активационная функция слоя;
- при обработке изображений применяются сверточные слои. Основной идеей является использование общих весов для нескольких нейронов и выборочная связность нейронов с предыдущим слоем (в отличии от

обычных, полносвязных слоев, в которых каждая компонента входного вектора подается на каждый нейрон слоя);

- вспомогательные слои, в которых вместо нейронов используется алгоритмическое преобразование входного вектора. Примерами являются слои нормализации, сокращения размерности (subsampling) и слой добавления шума (dropout). Использование таких модулей позволяет решить ряд существенных проблем, возникающих при обучении нейронной сети.

Примером сложной ИНС является GoogLeNet [13], схема которой приведена на рисунке 5. Эта нейронная сеть является одной из лучших в задаче классификации изображений и, несмотря на значительную сложность приведенного графа, полностью состоит из описанных выше слоев.

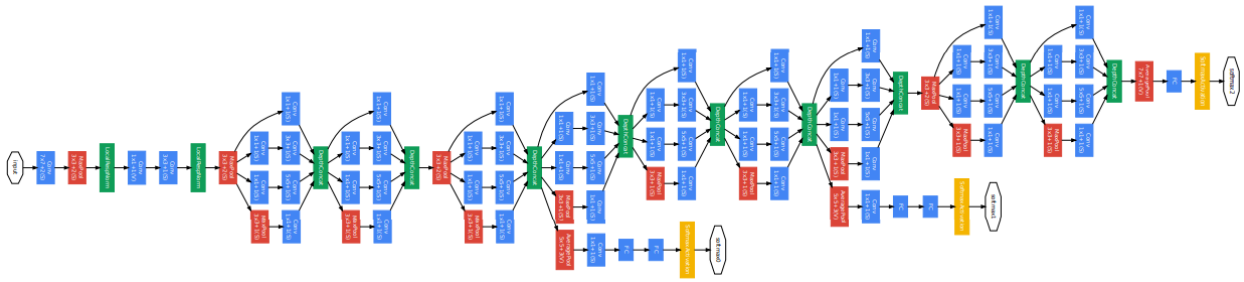

*Рис. 5.* Граф глубокой нейронной сети GoogLeNet [13].

Существуют различные подходы к классификации слоев нейронной сети. В данной работе будут использоваться следующие определения:

- вектор, который подлежит обработке нейронной сетью и подается на вход первому нейронному слою, обозначим как входной вектор;
- нейронный слой, стоящий в конце сети прямого распространения, выходом которого является результат обработки всей нейронной сети, обозначим как выходной слой;
- все нейронные слои, которые стоят перед выходным слоем, будем называть скрытыми.

Классической задачей, в которой нейронные сети превзошли другие алгоритмы, является распознавание рукописных цифр. Для настройки

и сопоставления методов распознавания изображений Национальным институтом стандартов и технологий США была составлена выборка рукописных цифр из 70000 изображений в градациях серого размером 28*28 пикселей [14]. В данной задаче наилучший результат показал ансамбль из нескольких нейронных сетей с ошибкой 0.23% неверно классифицированных изображений [15]. Для сравнения, лучший результат без использования нейронных сетей показал метод $k$ ближайших соседей с размером ошибки 0.52%.

# Постановка задачи

Выбор оптимальной архитектуры (глубина сети, ширина и тип слоев) ИНС представляется сложной задачей, решение которой обычно сводится к обучению множества нейронных сетей с различной архитектурой в поисках оптимальной. Даже для относительно простых нейронных сетей с одним скрытым слоем при определении его ширины используются крайне неэффективные методы. Целью данной работы ставится исследование этой проблемы и выявление численных характеристик нейронной сети, позволяющих на этапе обучения выявить избыток/недостаток нейронов в ИНС с одним скрытым слоем.

Описанное исследование будет проводиться в рамках задачи MNIST [14] на примере нейронной сети с одним скрытым слоем. Дополнительно в работе будет рассмотрена возможность переноса полученных результатов на другие задачи и архитектуры нейронной сети, а также возможность динамической оптимизации ширины скрытого слоя в процессе обучения.

# Проблематика и обзор литературы

Проблема выбора количества нейронов в скрытом слое не теряет свою актуальность со времен разработки первых нейронных сетей со скрытыми слоями и широко рассматривалась в работах 1990-х годов (напри-

мер [16, 17]). Основной целью выбора ширины слоя является избежание проблем underfitting и overfitting. При недостаточном количестве нейронов не все необходимые закономерности в данных будут выявлены нейронной сетью: эта проблема известна как underfitting и характеризуется большим значением функционала ошибки как на обучающих, так и на тестовых данных. При избыточном количестве нейронов может возникнуть проблема переобучения (overfitting), которая известна не только в области нейронных сетей, но и в статистике, и характеризуется маленьким значением функционала ошибки на обучающих данных и большим на тестовых. Вторым фактором при выборе количества нейронов является вычислительная сложность нейронной сети: при избыточной ширине требуется больше вычислительных ресурсов.

На интернет-форумах, посвященных проектированию нейронных сетей, широко распространена рекомендация по выбору количества нейронов в скрытом слое, которая, вероятно, берет свои истоки из работ [18, 19]: ширина скрытого слоя должна находиться в промежутке между размерностью входа слоя и количеством нейронов в следующем слое и быть близкой к $2/3$ размерности входа. Эта рекомендация, однако, имеет ряд недостатков, что и отмечается многими разработчиками. Основной проблемой таких рекомендаций является то, что они не учитывают особенности решаемой задачи, такие как сложность и размер обучающей выборки. Еще одним затруднением может являться то, что в многослойных нейронных сетях ширина последующего слоя также является величиной не фиксированной. Таким образом, предложенные рекомендации можно применять для выбора начального количества нейронов для последующей оптимизации более точными методами, один из которых будет рассмотрен в данной работе.

Несколько более точных подходов основаны на итеративном поиске оптимального количества нейронов путем динамического увеличения или уменьшения ширины скрытого слоя. Подход, при котором изначально нейронная сеть инициализируется заведомо избыточной шириной с последующим удалением малоактивных нейронов, называется прунингом (pruning). Этот подход основан на феномене отмирания малоактивных нейронов в

биологическом мозге и был рассмотрен в большом количестве работ (например, [17, 20]). В некоторых задачах такой подход позволяет находить оптимальное количество нейронов, однако начинать поиск оптимума «сверху» может быть вычислительно затратно. Также следует отметить, что многие из таких методов (например, [21]) для применения требуют изменение различных параметров нейронной сети (например, архитектуры, активационных функций), что затрудняет сравнение таких подходов с другими методами оптимизации ширины, а также может быть неприемлемо в некоторых задачах.

Другим подходом является поиск оптимума снизу путем дублирования наиболее активных нейронов, что предлагается в работе [16]. Однако применение предложенного метода может быть затруднено требованием значительного изменения архитектуры нейронной сети путем добавления дополнительных связей между нейронами слоя, что может оказаться неприемлемо во многих задачах. Методы динамического изменения ширины нейронного слоя заслуживают особого внимания, так как не требуют начинать процесс обучения заново после изменения архитектуры, в связи с чем будет рассмотрена возможность объединения преимуществ данных методов с разрабатываемыми в диссертации критериями для дальнейшего увеличения эффективности.

Также существуют работы, в которых предлагается выявлять переобучение с использованием дополнительных тестов. Это может быть периодическая проверка на тестовой выборке [22] или использование других видов тестирования [23]. Такие методы могут давать хорошие результаты, однако дополнительные тестирования могут быть вычислительно дорогими. В этом плане более эффективными могут оказаться методы, опирающиеся на информацию, доступную во время обучения, один из которых предлагается в данной работе.

Следует отметить, что помимо выбора архитектуры нейронной сети существуют и другие методы борьбы с переобучением. Примером таких приемов является dropout, нормализация и регуляризация весов и выходов нейронных слоев. Эти методы с успехом применяются в большинстве со-

временных глубоких сверточных нейронных сетей, и с их помощью удается избежать негативных эффектов избытка ширины слоя, но вычислительная сложность лишних нейронов остается.

# Глава 1

# Описание экспериментов

Для выполнения поставленной задачи произведено обучение ряда нейронных сетей с одним скрытым слоем. Количество нейронов в скрытом слое будет варьироваться в широком диапазоне, в то время как остальные параметры нейронной сети будут фиксированы. По итогам обучения для каждой НС будет вычислено значение функционала качества, а также ряд заранее определенных метрик. По значению функционала будет выявлено оптимальное значение ширины скрытого слоя, а также область значений, близкая к оптимальной. Затем полученные метрики будут анализироваться на наличие связи с оптимальным значением. В дальнейшем такой подход позволит определять оптимальность архитектуры нейронной сети по вспомогательным характеристикам (выявленным метрикам), доступным во время обучения.

Обучение будет проводиться на задаче MNIST, однако после выявления требуемых зависимостей будет рассматриваться возможность их применения в других задачах.

## 1.1 Выбор нейронной сети, алгоритма обучения и других параметров

С целью упрощения анализа результатов и уменьшения времени обучения нейронных сетей за основную архитектуру принята наиболее простая ИНС с одним скрытым слоем. Для решения задачи классификации рукописных цифр выходной слой будет содержать 10 нейронов. Активационной функцией этого слоя будет softmax (4), использование которой является распространенной практикой в задачах классификации. Такая функция позволяет интерпретировать величину выхода нейрона как вероятность отнесения исследуемого изображения к соответствующему классу (так как сумма активаций выходного слоя всегда равна единице). Обобщенная схема нейронной сети с одним скрытым слоем приведена на рисунке 1.1. Буквой $m$ на рисунке отмечено количество компонент входного вектора; $n$ — количество нейронов в скрытом слое, для этой величины также будет использоваться обозначение $N$; $p$ — количество выходных нейронов; $w_{ij}, i = 1, 2, ..., m, j = 1, 2, ..., n$ — элементы весовой матрицы $W$ скрытого слоя; $K_{i,j}, i = 1, 2, ..., n, j = 1, 2, ..., p$ — элементы весовой матрицы $K$ выходного слоя; $Y_1, Y_2, ..., Y_p$ — компоненты выходного вектора $\vec{y}$. В решаемой задаче $m = 28 * 28 = 784$ (количество входных пикселей), $p = 10$ (количество классов при классификации цифр).

Поскольку количество нейронов в выходном слое фиксировано, варьироваться может только ширина скрытого слоя $N$. В качестве активационной функции этого слоя используется сигмоида (3).

В современных нейронных сетях широко распространен подход, при котором перед применением активационной функции к взвешенной сумме нейрона прибавляется некоторое смещение (см. формулу 2). Величина этого смещения также является оптимизируемым параметром. Однако для уменьшения анализируемых параметров в исследуемой нейронной сети данный прием было решено не применять.

При решении задач классификации набор обучающих данных обычно состоит из пар $(\vec{x}, \vec{t})$, где $\vec{t} = (t_1, t_2, ..., t_p)$ — целевой вектор ($t$ от сло-

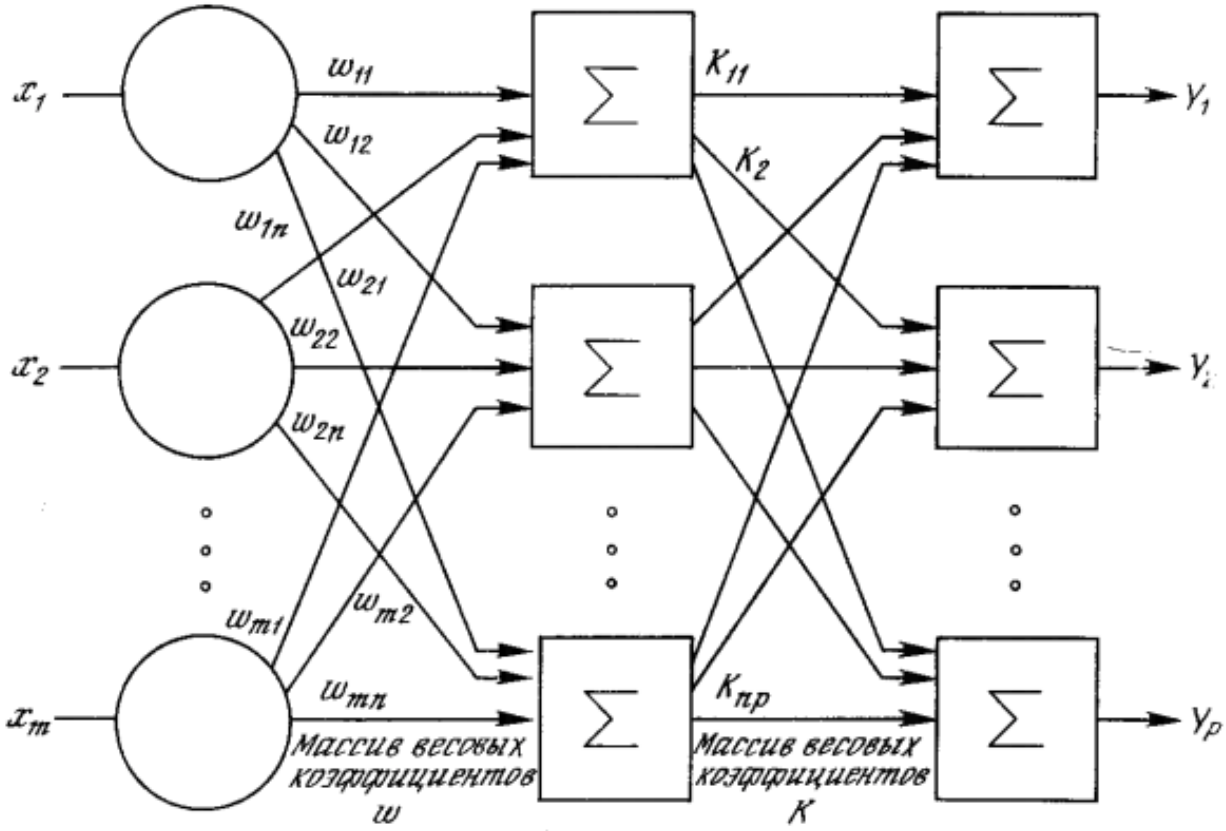


*Рис. 1.1.* Схема нейронной сети с одним скрытым слоем [7].

ва target), в котором закодирована информация о том, к какому классу относится соответствующее входное изображение $\vec{x}$. В частности, в задаче MNIST используется разреженное кодирование: все элементы $\vec{t}$ равны нулю кроме того, который соответствует цифре представленной на изображении (этот элемент равен единице).

Целью обучения описанной нейронной сети является найти такой набор весовых коэффициентов, чтобы в выходном слое активировался (выход нейрона был близок к единице) только тот нейрон, который соответствует цифре, представленной на входном изображении, в то время как остальные были неактивны (выход был близок к 0). Иначе эту задачу можно представить как минимизацию функционала ошибки

$$E_{l2}(\vec{t}, \vec{y}) = \sqrt{\sum_{i=1}^{p}(t_i - y_i)^2}. \tag{1.1}$$

Однако в задачах классификации часто используют в качестве функцио-

нала функцию перекрестной энтропии

$$H(\vec{t}, \vec{y}) = -\sum_{i=1}^{p} t_i \log y_i, \tag{1.2}$$

которая может способствовать более быстрому обучению. В рамках данной работы было решено использовать именно перекрестную энтропию.

Для обучения весов (минимизации ошибки) нейронной сети разработано множество методов, однако большинство из них основаны на методе градиентного спуска. Данный метод заключается в вычислении частных производных от функционала ошибки по каждому параметру и изменении этих переменных в направлении, обратном градиенту, с некоторым шагом. Величина этого шага равна модулю градиента, умноженному на специальный коэффициент, который в ИНС принято называть скоростью обучения. Во многих задачах обучения нейронных сетей хорошо себя зарекомендовали модифицированные алгоритмы градиентного спуска, такие как спуск с моментом или адаптивная оптимизация (например алгоритмы adam, adagrad [24, 25]), однако в данной работе для простоты анализа результатов будет использоваться классический градиентный спуск.

Выбор эффективной скорости обучения является отдельной нетривиальной задачей. Значение этого коэффициента может быть постоянным или варьироваться в процессе обучения. В частности, распространен подход, при котором его значение уменьшается со временем, стремясь к нулю. В рамках рассматриваемой задачи сравнения различных архитектур этот коэффициент принят постоянным и равным единице, так как это значение показало хорошие результаты при проведении вспомогательных тестов.

Следует отметить, что в процессе обучения итерационно оптимизируется не функционал (1.2) для одного изображения, а среднее значение этого функционала по всей обучающей выборке. Однако для оценки обобщающих способностей нейронной сети после обучения требуется проводить проверку ее качества на данных, не участвовавших в обучении (тестовой выборке). Явление, при котором уровень ошибки на тестовых данных значительно превосходит ошибку на обучающей выборке, называется переобу-

чением (overfitting) и свидетельствует о низком качестве обученной НС. Проблема переобучения распространена при оптимизации нейронных сетей, и множество методов нацелены на борьбу с ней: слои dropout, искусственное расширение обучающей выборки (augmentation), ограничение роста весов (regularization and normalization) и др. Одной из причин переобучения может являться избыток нейронов, что будет рассмотрено далее в этой работе. Разработчики MNIST предоставляют набор данных уже разделенным на два множества: 60000 обучающих изображений и 10000 тестовых. Именно среднее значение функционала (1.2) по всем изображениям тестового множества является главным показателем качества нейронной сети, и по нему будет производится сравнение различных нейронных сетей.

Еще одним фактором, от которого зависит процесс обучения, является начальная инициализация значений весов. На данный момент широко распространено применение инициализации Глорота [26]. Следует отметить, что данный вид инициализации особенно эффективен при обучении глубоких нейронных сетей, так как помогает решить проблему затухающих градиентов. Его применение в коротких нейронных сетях также допустимо, однако в данной работе будет использоваться более простая инициализация весов случайными значениями, равномерно распределенными в интервале $[-1, 1]$.

Для ускорения обучения будет применяться распространенный подход, при котором на вход нейронной сети изображения подаются не по одному, а группами (батчами). В глубоких нейронных сетях такой подход имеет большое значение, так как позволяет эффективно контролировать рост весов (batch normalization [27]), однако в коротких нейронных сетях влияние на качество обучения меньше и используется в основном для оптимизации использования оперативной памяти. Исходя из ограничений, накладываемых используемой для вычислений инфраструктурой, был выбран размер батча в 10000 изображений.

В зависимости от решаемой задачи, выбранной архитектуры, скорости обучения и других параметров для оптимизации нейронной сети требу-

ется различное количество итераций обновления весов градиентным методом. В подавляющем большинстве случаев требуемое количество итераций сильно превосходит количество пар в обучающей выборке. В связи с этим широко распространен подход, при котором каждое обучающее изображение предъявляется (подается на вход) нейронной сети несколько раз. Процесс однократного предъявления всех изображений обучающей выборки принято называть эпохой. Например, обучение сверточной нейронной сети, показавшей лучший результат на задаче mnist, заняло более 500 эпох [15], т.е. каждое из 60000 обучающих изображений было использовано не менее 500 раз. В рассматриваемой задаче оптимизации ширины нейронной сети обучение производится в течение 100 эпох, так как за это время обучение большинства рассматриваемых НС сходится, и ошибка практически перестает уменьшаться.

## 1.2 Выбор метрик

Целью данной работы является выявление признаков, по которым во время обучения возможно выявить избыток или недостаток нейронов. Для этого производится анализ параметров, доступных во время обучения, и их связи с итоговым значением функционала качества. В качестве таких параметров рассматриваются:

- весовая матрица $W$ нейронов скрытого слоя размерности $[m \times n]$;
- весовая матрица $K$ нейронов выходного слоя размерности $[n \times p]$;
- активация нейронов $Y$ скрытого слоя на тестовой выборке. Размерность этой матрицы будет $[L \times n]$, где $L = 10000$ — количество тестовых изображений.

Можно заметить, что одна из размерностей каждой рассматриваемой матрицы переменная. Для удобства при анализе будут сравниваться строки матриц $W^T, K, Y^T$, так как их строки будут иметь фиксированную длину.

Эти матрицы будут рассматриваться, так как было сделано предположение, что при увеличении количества нейронов в скрытом слое $n$ сходство строк этих матриц будет расти. Для оценки этого сходства помимо описанных матриц будут рассматриваться производные от них матрицы:

- матрицы попарных евклидовых расстояний между строками исходных матриц;
- матрицы попарных корреляционных расстояний между строками исходных матриц. Под корреляционным расстоянием понимается величина

$$1 - r(\vec{x}, \vec{y}), \tag{1.3}$$

где $r(\vec{x}, \vec{y})$ — коэффициент корреляции Пирсона векторов $\vec{x}$ и $\vec{y}$.

Вместе с описанными матрицами будут рассматриваться некоторые виды их нормировки:

- линейное масштабирование всех элементов матрицы в промежуток от -1 до 1;
- линейное масштабирование строк рассматриваемых матриц в промежуток от -1 до 1 независимо друг от друга;
- нормализация всех элементов матрицы таким образом, чтобы их среднее значение равнялось нулю, а стандартное отклонение единице;
- аналогичная независимая нормализация всех строк рассматриваемых матриц.

Итого будет рассматриваться 45 различных матриц. Для удобства анализа эти матрицы будут редуцированы в скалярную величину одним из следующих образов:

- найден минимальный элемент матрицы;
- найден максимальный элемент матрицы;

- найдено среднее значение элементов матрицы;
- некоторые другие методы.

Всего будет анализироваться 225 параметров путем построения графиков зависимости от количества нейронов в скрытом слое $n$.

# Глава 2

# Программная реализация, тестирование и анализ результатов

Для эффективного проведения большого количества экспериментов было решено использовать высокопроизводительную библиотеку для машинного обучения Tensorflow [28] с поддержкой вычислений на графическом процессоре (GPU). Поддержка GPU вычислений основана на библиотеках CUDA [29] и CUDNN [30]. Tensorflow была выбран в связи с наличием открытого исходного кода и подробной документации. Предоставленные в документации рекомендации, а также наличие API разного уровня абстракции позволяет писать приложения различной сложности, максимально эффективно использующие предоставленные вычислительные ресурсы.

## 2.1 Инфраструктура

Для проведения экспериментов использовался компьютер и программное обеспечение со следующими характеристиками:

**CPU** Intel Pentium N4200, 1.10GHz * 4 потока;

**доступная ОЗУ** 2.7 GB;

**GPU** Nvidia GeForce 810M, 0.77GHz * 96 ядер CUDA, 2GB выделенной видео памяти;

**Tensorflow** версия 1.3, precompliled with GPU support;

**CUDA** версия 8.0;

**CUDNN** версия 6.0.

Несмотря на использование процессора с устаревшими характеристиками, удалось быстро провести большое количество экспериментов за счет эффективного использования GPU. Всего в рамках основной серии экспериментов было обучено 273 нейронных сети за 35 часов, обработка полученных результатов заняла 30 часов (при этом использовалось только вычисление на CPU).

Как отмечалось выше, ограничение видеопамяти в 2 Gb ограничило размер батча до 10000. Так как в процессе экспериментов приходилось обрабатывать матрицы больших размерностей, доступная оперативная память позволила использовать не более 9000 нейронов в скрытом слое.

## 2.2 Результаты тестирования, выбор метрик

Было проведено 273 эксперимента в диапазоне от 2 до 8709 нейронов в скрытом слое. Логарифмический график зависимости ошибки ($loss$) на тестовом множестве от количества нейронов ($N$) приведен на рисунке 2.1, откуда видно, что оптимальным оказалось 575 нейронов в скрытом слое. В диапазоне от 223 до 575 нейронов $loss < 0.24$, поэтому этот интервал также будет считаться оптимальным.

На 15 из 225 графиков были замечены некоторые особенности в области оптимальных значений N, которые в дальнейшем позволят выявить по значению исследуемого параметра недостаток или избыток нейронов. Одним из видов выявленных зависимостей является асимптотическое стрем-

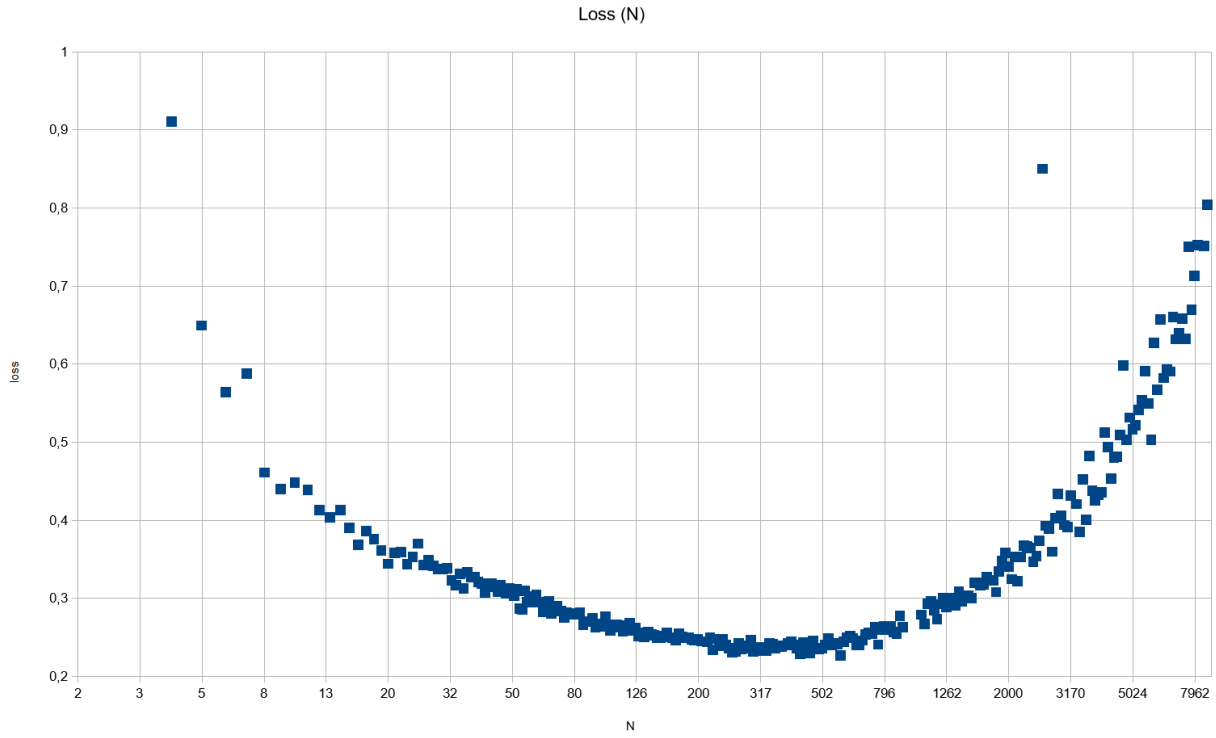


*Рис. 2.1.* Зависимость среднего значения функционала ошибки (1.2) по тестовой выборке от количества нейронов в скрытом слое.

ление исследуемой величины $f(N)$ к некоторому постоянному значению $a$ при росте $N$. Таким образом, если выявить максимальное отклонение $\varepsilon$ в области оптимальных значений, то при $|f(N) - a| >> \varepsilon$ можно сделать вывод о недостатке нейронов. Показательным примером этого класса функций является величина $f_1(N)$, где $f_1$ — среднее значение элементов матрицы попарных расстояний между строками $W^T$. В качестве метрики берется корреляционное расстояние (1.3). Зависимость $f_1(N)$, для которой $a = 1$ и $\varepsilon = 0.004$, приведена на графике 2.2.

Другим видом найденных зависимостей являются функции с экстремумом в области оптимальных значений. Такие функции позволяют выявить не только недостаток нейронов, но и их избыток, однако полученные измерения параметров такого типа имеют большой разброс значений при близких $N$, а экстремум выражен не четко (со стороны больших значений $N$ производная близка к нулю). Лучшим примером параметров такого типа является $f_2(N)$. $f_2$ — среднее значение элементов матрицы попарных евклидовых расстояний между строками матрицы $K$ (график 2.3). Минимум достигается при $N = 389$ и равен 2.31. Следует отметить, что ось

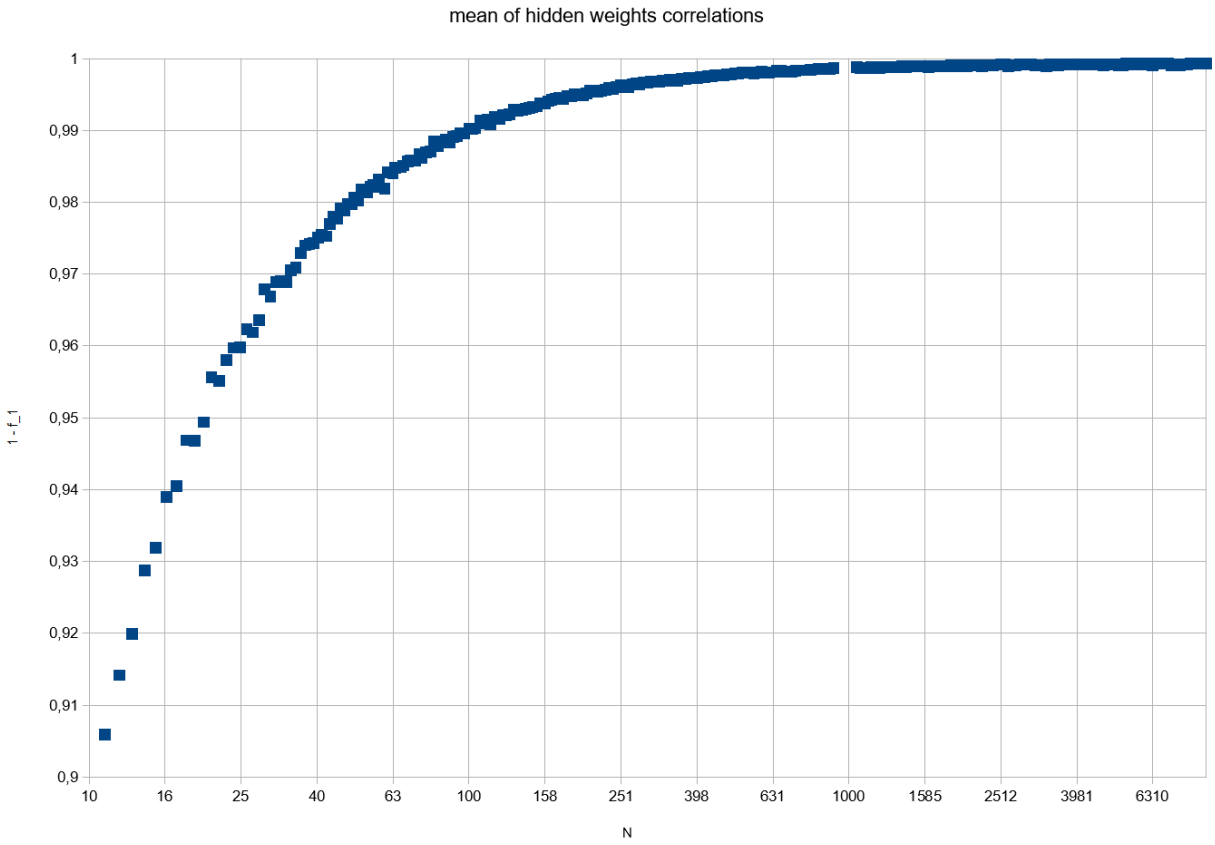


*Рис. 2.2.* Зависимость параметра $f_1$ от количества нейронов в скрытом слое.

X графика 2.3 является логарифмической, из-за чего может складываться обманчивое впечатление выраженности экстремума.

## 2.3 Эволюция выбранных параметров во время обучения

Следует отметить, что приведенные на графиках значения вычислены после обучения нейронных сетей на протяжении 100 эпох. Большую практическую пользу удастся извлечь, если с помощью рассматриваемых величин выявлять оптимальность ширины слоя на более ранних эпохах. Для этого требуется рассмотреть эволюцию этих параметров во время обучения.

На графике 2.4 можно наблюдать изменение величины $f_1$ в процессе обучения для некоторых значений ширины скрытого слоя. Для удобства на графике приведена величина $1-f_1$ с логарифмической шкалой. Из графика

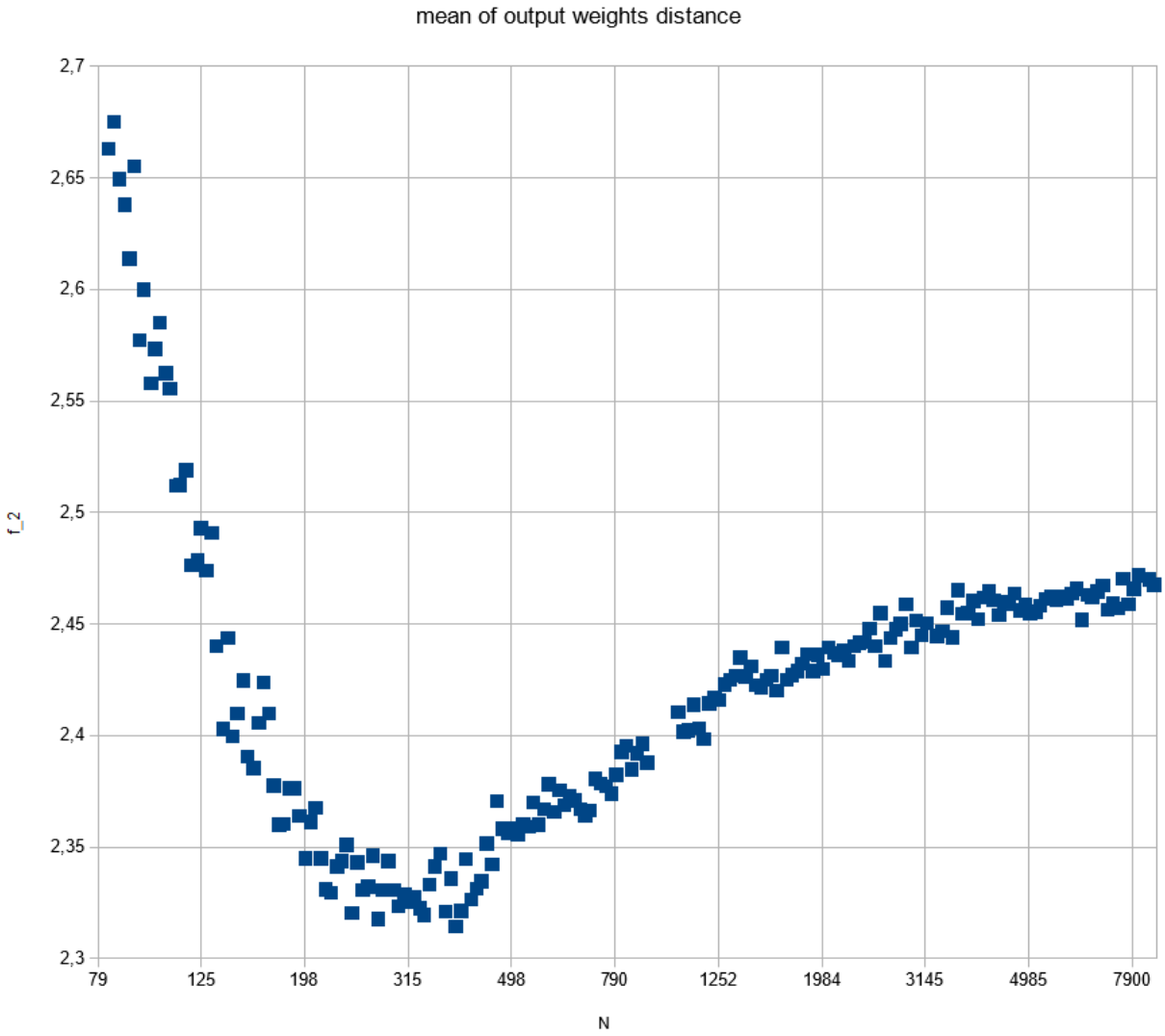


*Рис. 2.3.* Зависимость параметра $f_2$ от количества нейронов в скрытом слое.

видно, что величина $f_1$ мало изменяется на протяжении всего обучения и может указать на нехватку нейронов уже на ранних этапах обучения (при значительном отклонении от оптимального порога $\varepsilon$).

На графиках 2.5 и 2.6 представлена эволюция параметра $f_2$ для некоторых значений ширины скрытого слоя в разных масштабах. Из графика 2.5 видно, что на ранних этапах обучения (до 20-й эпохи) по абсолютному значению $f_2$ сложно выявить недостаток нейронов, так как $f_2(N)|_{N\leqslant 100} < f_2(251)$. Однако недостаток нейронов можно выявить, если наблюдать за изменением $f_2$ в процессе обучения: при малых $N$ его производная положительна в отличие от $f_2(251)$. На графике 2.6 можно заметить, что уже на начальных этапах обучения значение $f_2$ при оптимальных значениях $N$

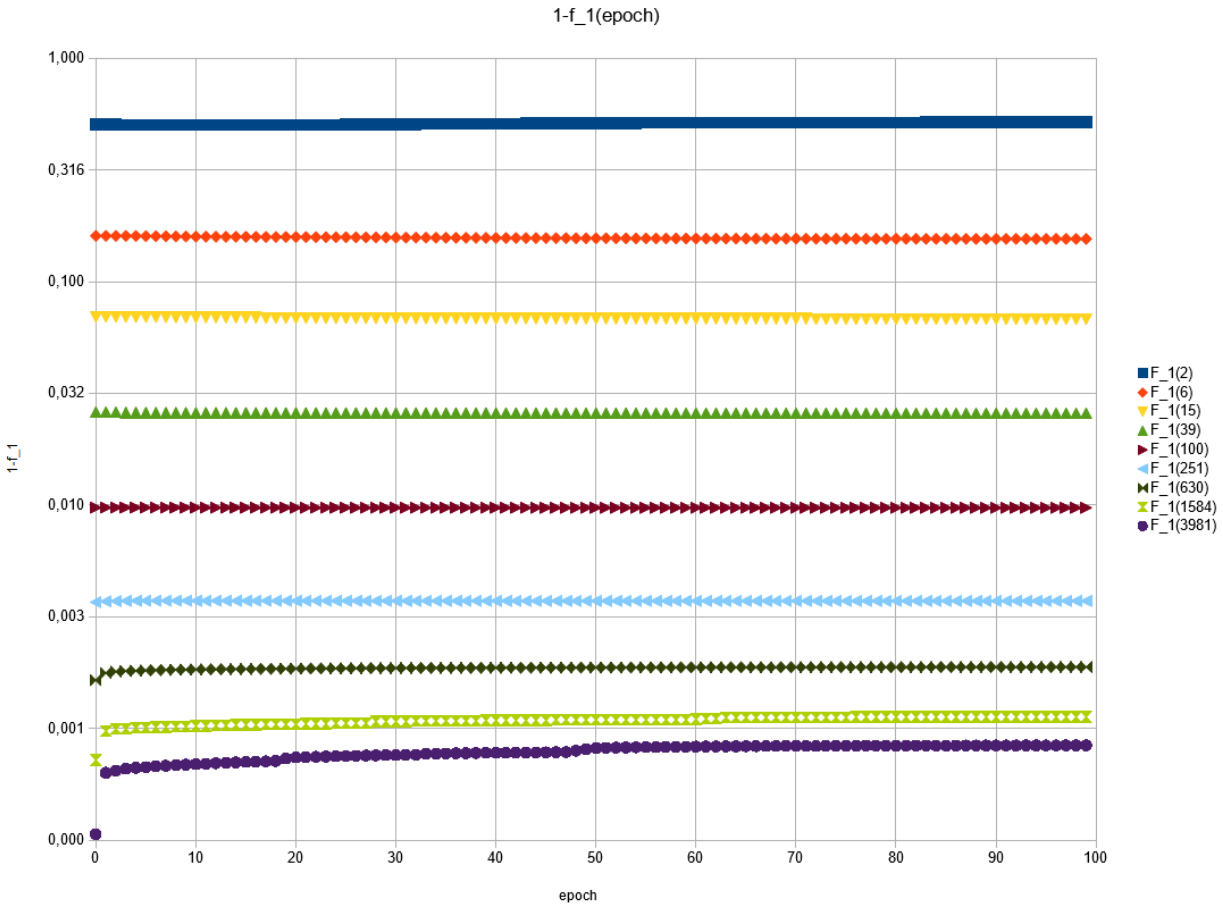


*Рис. 2.4.* Эволюция параметра $f_1$.

меньше, чем при избытке нейронов. Таким образом, даже после нескольких эпох обучения можно выявить избыток ширины скрытого слоя, однако для более точных результатов следует провести больше эпох: на графике можно видеть, что $N = 1584$ (более близкое к оптимальному значению чем $N = 3981$) до 20 эпохи имеет значение $f_2(1584) > f_2(3981)$.

## 2.4 Анализ результатов

Результатом анализа параметров стали два типа зависимостей, которые позволят во время обучения выявить нехватку нейронов. С помощью параметров второго типа возможно также выявление избытка нейронов, однако это может быть затруднено из-за большого разброса параметров при близких значениях $N$ и маленького наклона функции в области больших значений $N$.

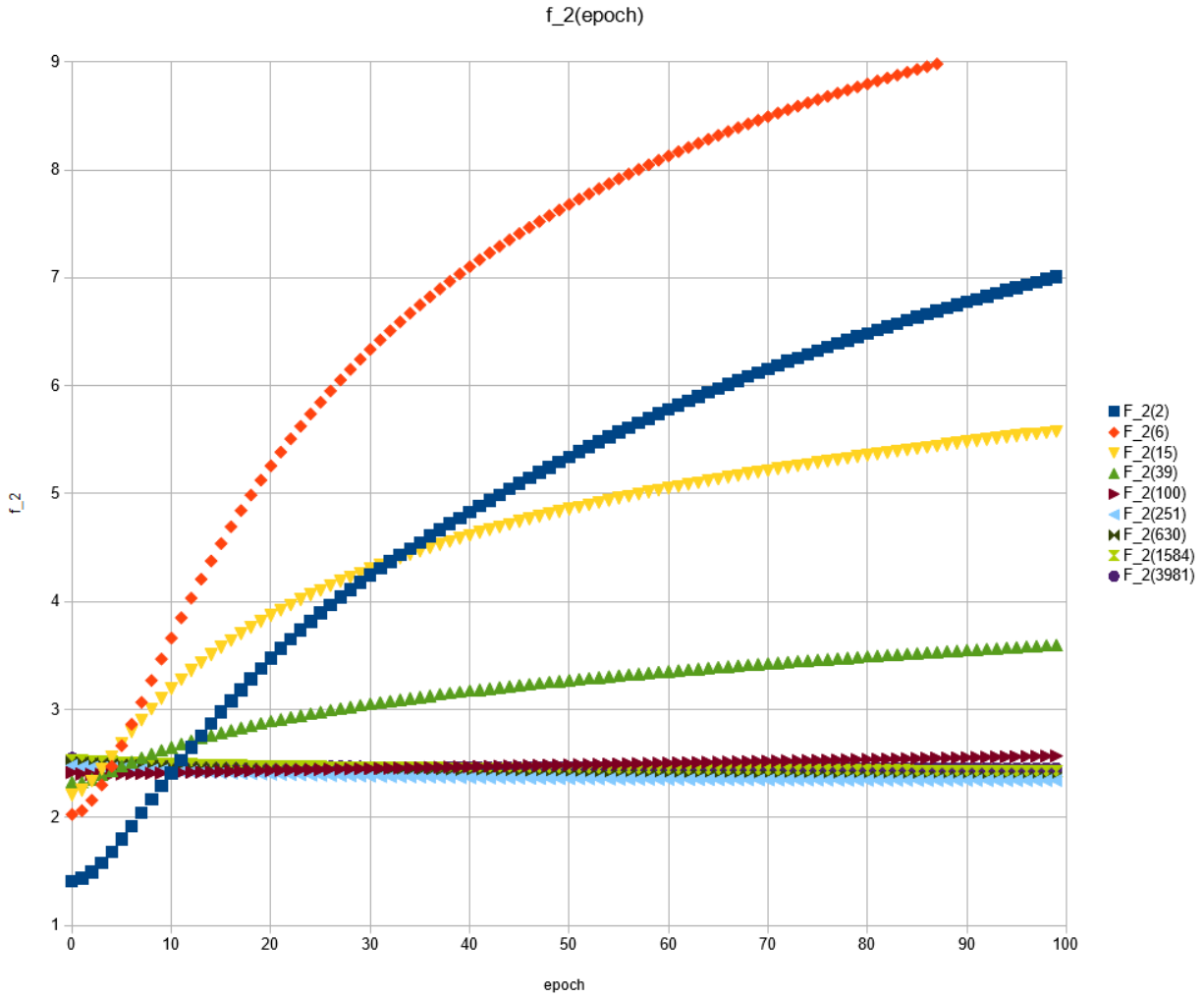


*Рис. 2.5.* Эволюция параметра $f_2$.

Преимуществом функций с асимптотой является возможность выявления значительного недостатка нейронов по абсолютному значению параметра, однако надежность этого критерия требует дополнительного исследования, так как значение $\varepsilon$ может сильно варьироваться от задачи к задаче. При решении новых задач значение $\varepsilon$ будет неизвестно, что делает невозможным применение критерия $f_1$ в большинстве новых задач.

Более перспективным представляется исследование функций с экстремумом, так как может выявить и избыток нейронов, однако для нахождения экстремума требуется вычисление в двух точках (т.е. нужно обучать две сети с разным количеством нейронов). При выборе этих точек следует учитывать амплитуду разброса значений этого параметра и малый наклон функции в области больших $N$.

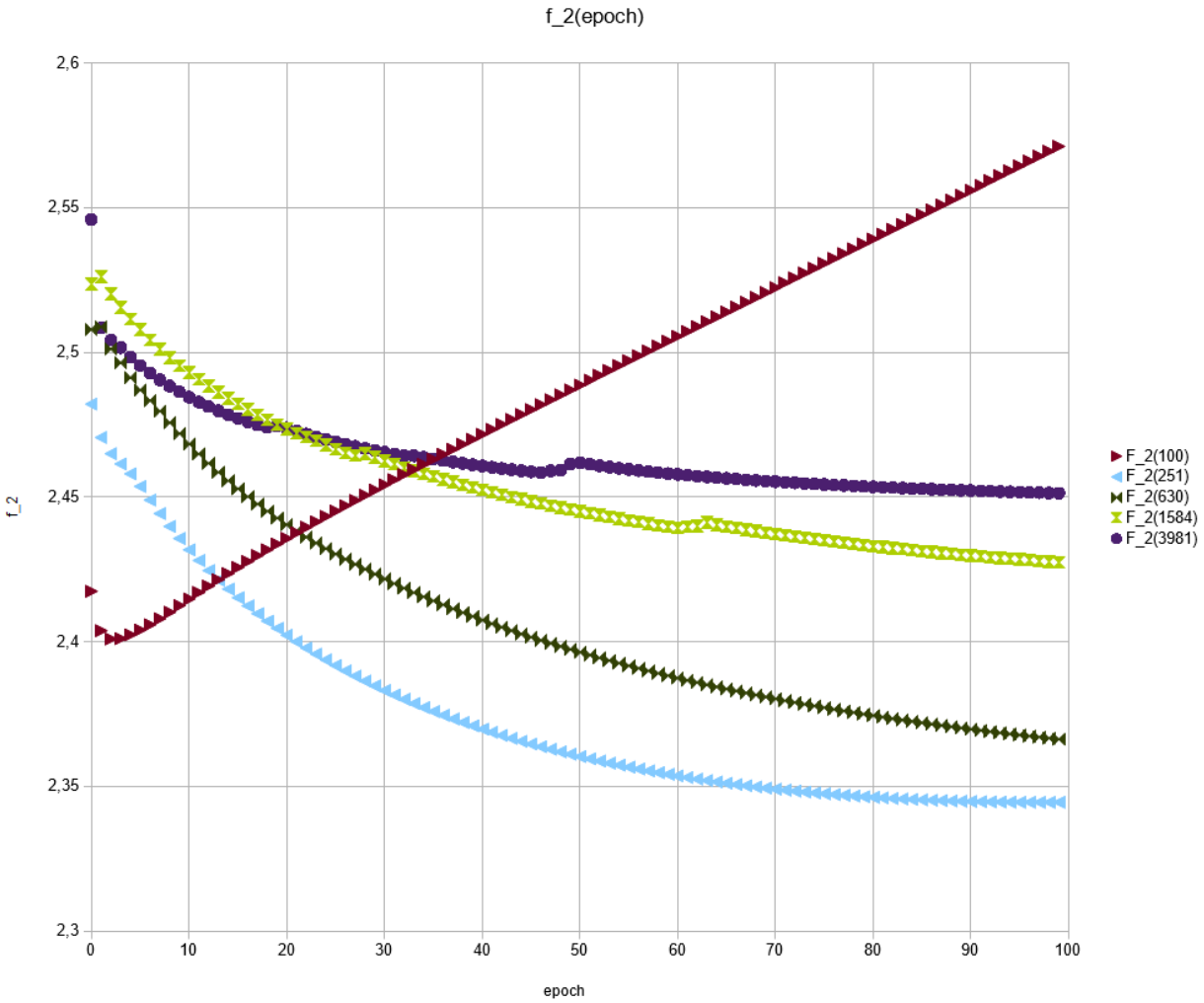


*Рис. 2.6.* Эволюция параметра $f_2$ (увеличенный масштаб).

# Глава 3

# Применение разработанных критериев и перспективы

Описанные в предыдущей главе критерии могут быть применены для выявления недостатка нейронов одним из следующих образов:

- по абсолютному значению критерия $f_1$ в случае известного значения $\varepsilon$;
- по динамике изменения величины $f_2$ в процессе обучения: ее возрастание со временем указывает на недостаток нейронов;
- по значению $f_2$ при двух значениях $N$: количество нейронов с меньшим значением $f_2$ ближе к оптимуму, однако при выборе проверяемых $N$ следует учитывать величину разброса $f_2$ при близких значениях $N$, которая может варьироваться в зависимости от задачи. Также следует учитывать, что перед применением данного метода должно пройти некоторое количество итераций обучения (для рассматриваемой задачи около 40 эпох). Эти ограничения делают данный метод менее приоритетным.

В рамках предложенных методов выявление избытка нейронов возможно только по значению $f_2$ в двух точках. При выборе этих точек следует учитывать разброс и малое значение производной при больших $N$ (величина производной может варьироваться в зависимости от задачи), а также для

более точных результатов может потребоваться предварительное обучение (в рассматриваемой задаче около 30 эпох).

Разработанные методы могут применяться для оценки оптимальности ширины нейронных сетей с одним скрытым слоем и значительного сокращения количества итераций перебора архитектур. Однако следует учитывать, что при значительном изменении параметров нейронной сети критерии оптимальности могут измениться, и может потребоваться дополнительное исследование новой нейронной сети с использованием описанной методологии. Примерами таких изменений могут быть изменение активационных функций, функционала ошибки, методов оптимизации или применение bias.

## 3.1 Динамическое расширение слоя

Описанные в предыдущей главе критерии позволяют выявить недостаток нейронов, позволяя более точно выбрать ширину скрытого слоя, не ожидая окончания обучения. При этом время экспериментов может значительно сокращаться, даже если при изменении ширины слоя обучение начинать сначала. Однако лучших результатов возможно достичь при использовании методов динамического изменения ширины слоя без необходимости начинать процесс обучения заново. Пример таких алгоритмов рассмотрен в работах [17] для динамического сужения слоя и [16] для расширения. При таком подходе удастся достигать оптимальной архитектуры и наименьшего значения функционала, полностью отказавшись от перебора параметров.

## 3.2 Многослойные сети

На данный момент большинство сложных задач решаются глубокими нейронными слоями с большим количеством слоев. Хотя возможность применения предложенных критериев в многослойных, а также сверточ-

ных нейронных сетях требует дополнительного исследования, описанная методология исследований может быть полезной при выборе новых критериев для глубоких нейронных сетей. В случае успеха удастся индивидуально выбирать количество нейронов каждого слоя. При этом высокую практическую ценность будут нести даже те критерии, которые указывают на неоптимальность слоя только после полного окончания обучения (по завершении выбранного количества эпох) нейронной сети из k слоев, так как это на k порядков сократит количество итераций перебора за счет параллельной оптимизации каждого слоя.

# Выводы

В результате проведенного исследования было предложено несколько критериев для оценки качества архитектуры ИНС с одним скрытым слоем. В частности, таким информативным параметром стали средние значения матриц попарных расстояний (в разных метриках) весовых векторов скрытого и выходного слоев, с помощью которых возможно выявить недостаток нейронов, а в некоторых случаях и избыток.

Эти параметры удалось выявить в результате проведения экспериментов при различных значениях ширины скрытого слоя. По результатам этого тестирования была определена область оптимального количества нейронов и среди более чем 200 различных параметров производился поиск тех, которые в области оптимальных значений имеют некоторую особенность. Примером такой особенности стал экстремум функции. Менее выраженной особенностью является близость параметра к некоторой асимптоте в области оптимальных значений.

Было показано, что сделать вывод об оптимальности архитектуры можно не только после окончания, но и на более ранних этапах обучения. Хотя возможность применения данных критериев имеет ряд ограничений (рассмотренных в работе), в некоторых задачах использование предложенных характеристик позволит значительно увеличить эффективность процедуры поиска оптимального количества нейронов.

Рассмотрены пути дальнейшего развития методов выбора оптимальной архитектуры, среди которых объединение с методами динамического изменения ширины слоев, не прерывающих процесс обучения, а также применение в глубоких и сверточных нейронных сетях.

# Заключение

В работе предложены несколько критериев, с помощью которых возможно выбрать ширину нейронного слоя близкую к оптимальной на начальных этапах обучения ИНС. Предложенные методы позволяют в неко-

торых задачах значительно укорить процесс выбора архитектуры нейронных сетей с одним скрытым слоем, а также открывают широкие возможности для исследований проблемы выбора оптимальной архитектуры на более сложных сетях: многослойных и сверточных.

# Литература


[1] Hinton G. E., Osindero S., Teh Y.-W. A fast learning algorithm for deep belief nets // Neural Computation. 2006. V. 18, issue 7. P. 1527–1554.

[2] LeCun Y., Bottou B., Orr G. B., Muller K.-B. Efficient BackProp // Neural Networks: Tricks of the Trade. 1996. P. 9–50.

[3] AI winter // Wikipedia URL: https://en.wikipedia.org/wiki/AI_winter (дата обращения: 05.05.2018).

[4] He K., Zhang X., Ren S., Sun J. Deep residual learning for image recognition // IEEE Conference on Computer Vision and Pattern Recognition. 2016. P. 770–778.

[5] McCulloch W. S., Pitts W. A logical Calculus of Ideas Immanent in Nervous Activity // The bulletin of mathematical biophysics. 1943. P. 115–133.

[6] Rosenblatt F. The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain // Psychological Review. 1958. V. 65. No. 6. P. 386–408.

[7] Уоссермен Ф. Нейрокомпьютерная техника. М.: Мир, 1992. 184 с.

[8] Хайкин С. Нейронные сети: полный курс. 2-е изд. М.: Вильямс, 2008.

[9] Carpenter G., Grossberg S. A Massively Parallel Architecture for a Self-Organizing Neural Pattern Recognition Machine // Computer vision, graphics, and image processing. 1987. No. 37. P. 54–115.

[10] Ращенко Д. В. Модификация нейронной сети АРТ-1, направленная на сокращение фазы поиска // Процессы управления и устойчивость. 2016. Т. 3. № 1. С. 477–481.

[11] Karpathy A., Fei-Fei L. Deep visual-semantic alignments for generating image descriptions // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2017. V. 39. P. 664–676.

[12] Rashchenko D. V. Elimination of the search phase in the neural network ART-1 by changing the criterion of vectors similarity // IEEE 2015 International Conference «Stability and Control Processes» in memory of V. I. Zubov (SCP). 2015. P. 661–662.

[13] Szegedy C., Liu W., Jia Ya., Sermanet P., Reed S., Anguelov D., Erhan D., Vanhoucke V., Rabinovich A. Going deeper with convolutions // IEEE Conference on Computer Vision and Pattern Recognition. 2015. P. 1–9.

[14] THE MNIST DATABASE of handwritten digits // Yann LeCun URL: http://yann.lecun.com/exdb/mnist/index.html (дата обращения: 06.05.2018).

[15] Ciresan D., Meier U., Schmidhuber J. Multi-column deep neural networks for image classification // IEEE Conference on Computer Vision and Pattern Recognition. 2012. P. 3642–3649.

[16] Fahlman S. E., Lebiere C. The cascade-correlation learning architecture // D. S. Touretzky, Advances in neural information processing systems 2. San Francisco: Morgan Kaufmann, 1990. P. 524–532.

[17] Psichogios D. C., Ungar L. H. SVD-NET: an algorithm that automatically selects network structure // IEEE Transactions on neural networks. 1994. V. 5. P. 513–515.

[18] Boger Z., Guterman H. Knowledge extraction from artificial neural networks model // IEEE Systems Man and Cybernetics-Compulational Cybernetics and Simulation. 1997. V. 4. P. 3030–3035.

[19] Blum A. Neural networks in C++: an object-oriented framework for building connectionist systems. V. 1. New York: John Wiley & Sons, 1992. 224 p.

[20] Manessi M., Rozza A., Bianco S., Napoletano P., Schettini R. Automated Pruning for Deep Neural Network Compression // arXiv:1712.01721. 2017.

[21] Fnaiech F., Fnaiech N., Najim M. A new feedforward neural network hidden layer neuron pruning algorithm // IEEE International Conference on Acoustics, Speech, and Signal Processing. 2001. P. 1277–1280.

[22] Shin-ike K. A two phase method for determining the number of neurons in the hidden layer of a 3-Layer neural network // The Society of Instrument and Control Engineers (SICE) Annual Conference. 2010. P. 238–242.

[23] Rivals I., Personnaz L. A statistical procedure for determining the optimal number of hidden neurons of a neuralmodel // Second International Symposium on Neural Computation. 2000.

[24] Kingma D. P., Ba J. Adam: a method for stochastic optimization // International Conference on Learning Representations. 2015.

[25] Duchi J., Hazan E., Singer Y. Adaptive subgradient methods for online learning and stochastic optimization // The Journal of Machine Learning Research. 2011. V. 12. P. 2121–2159.

[26] Glorot X., Bengio Yo. Understanding the difficulty of training deep feedforward neural networks // Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. 2010. P. 249–256.

[27] Ioffe S., Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift // Proceedings of The 32nd International Conference on Machine Learning. 2015. P. 448–456.

[28] Tensorflow URL: https://www.tensorflow.org/ (дата обращения: 05.05.2018).

[29] Параллельные вычисления CUDA // NVIDIA URL: http://www.nvidia.ru/object/cuda-parallel-computing-ru.html (дата обращения: 05.05.2018).

[30] NVIDIA cuDNN GPU Accelerated Deep Learning // NVIDIA Developer URL: https://developer.nvidia.com/cudnn (дата обращения: 05.05.2018).